

Lumidora

v1

Lumidora: Eine innovative Open-Source KI-Plattform

Lumidora ist ein Open-Source-Projekt, das verschiedene Open-Source-Technologien kombiniert, um vielseitige Anwendungen zu ermöglichen. Hier ist eine klare Übersicht:

Funktionen und Anwendungen

Chatbot: Lumidora ermöglicht lebendige Gespräche dank natürlicher Sprachverarbeitung.

Content Creator: Lumidora erstellt automatisch Texte, Artikel, Geschichten und Gedichte.

Text-to-Speech (TTS): Lumidora verwandelt Texte in lebenssechte Sprachausgabe.

Speech-to-Text (STT): Lumidora verwandelt gesprochene Worte in Text.

Text-to-Video: Lumidora verwandelt Texte in Videos.

Open Source Tools: Lumidora integriert verschiedene Open Source-Technologien und kann angepasst werden.

Technologie und Architektur

Lumidora nutzt Open Source-Technologien, um natürliche Sprache zu verstehen und menschenähnliche Interaktionen zu ermöglichen.

Installation und Nutzung

Die Dokumentation bietet Anleitungen zur Installation und Konfiguration von Lumidora auf verschiedenen Plattformen sowie Integrationsschritte in Projekte und Anwendungen.

Fazit

Lumidora vereint verschiedene Open Source-Projekte, um die Interaktion mit Technologie und die Erstellung von Inhalten zu revolutionieren. Wir laden Entwickler, Kreative und Technologiebegeisterte ein, Lumidora zu erkunden und zu nutzen, um die Zukunft der KI mitzugestalten. Willkommen in der Welt von Lumidora, wo Innovation und Offenheit Standard sind.

Prinzipien

Allgemeine Prinzipien

- Für dieses Projekt wird nur Open Source Software verwendet.
- Technische Vielfalt wird kontrolliert um Komplexität zu reduzieren
- Einfache Lösungen: Die Lösung muss so einfach wie möglich sein. Gibt es Komplexität, wird diese möglichst hinter einem einfachen Interface versteckt.
- Patente, Copyrights und Geheimnisse werden geschützt und berücksichtigt

Daten

- Informationsoffenheit – Informationen müssen offen und verfügbar sein, um Produktivität und Innovation zu unterstützen
- Es sollte möglichst mit aktuellen Daten und Models gearbeitet werden

Serviceorientierte Architektur

- Aufgabeneinteilung (Separation of Concerns) – Es wird möglich sein, eine Komponente mit minimalen Auswirkungen auf andere Komponenten zu ändern

Benutzerfreundlichkeit

- Benutzerinteraktion muss maximal einfach und maximal nützlich sein. Der Assistent soll den Benutzer bei seiner Arbeit schnell, effizient und einfach dienlich sein.

Sichten

Systemkontext

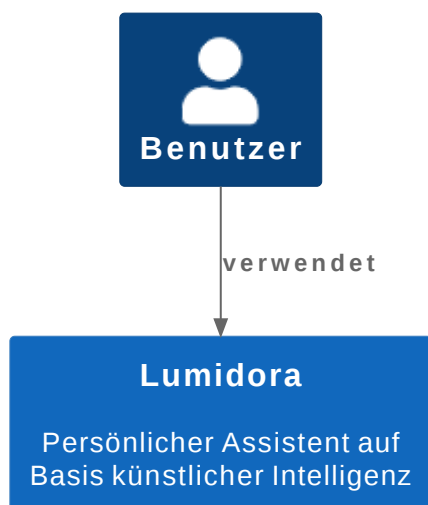
Das nachfolgende Schaubild zeigt das Zusammenspiel von Lumidora mit der näheren Umwelt. Es ist eine Vogelperspektive und zeigt eine Gesamtübersicht der Verbindungen und Schnittstellen von und zu Lumidora. Lumidora wird als Blackbox gezeigt und erst in weiteren Diagrammen näher betrachtet.

Notiz: Funktional mit UML-Klassendiagramm. Technisch mit Verteilungsdiagramm (Deployment Diagram)

Funktionaler Kontext

Entität	Beschreibung
Benutzer	Jeder Benutzer, welcher mit Hilfe von KI seine Arbeit mit dem Computer optimieren will. Benutzer (Content-Creator), welche mithilfe von Lumidora Kontent erstellen möchten. Zum Beispiel Youtuber, Influencer. Softwareentwickler, welche mithilfe von Lumidora ihren Softwareentwicklungsprozess optimieren möchten.
Lumidora	Persönlicher Assistent auf Basis künstlicher Intelligenz.

Systemkontext Funktional "Was"



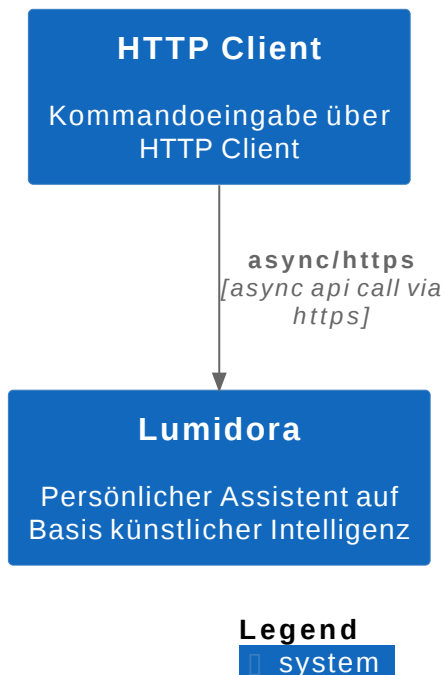
Legend

person
system
container
component
external person
external system
external container
external component

Technischer Kontext

Entität	Beschreibung
HTTP Client	HTTP Client jeglicher Art, kann auch ein oder mehrere Web-Frontends für menschliche Benutzer darstellen. Kann aber auch ein technischer HTTP Client sein, der von anderen Anwendungen aus das Lumidora System aufruft.
Lumidora	Persönlicher Assistent auf Basis künstlicher Intelligenz

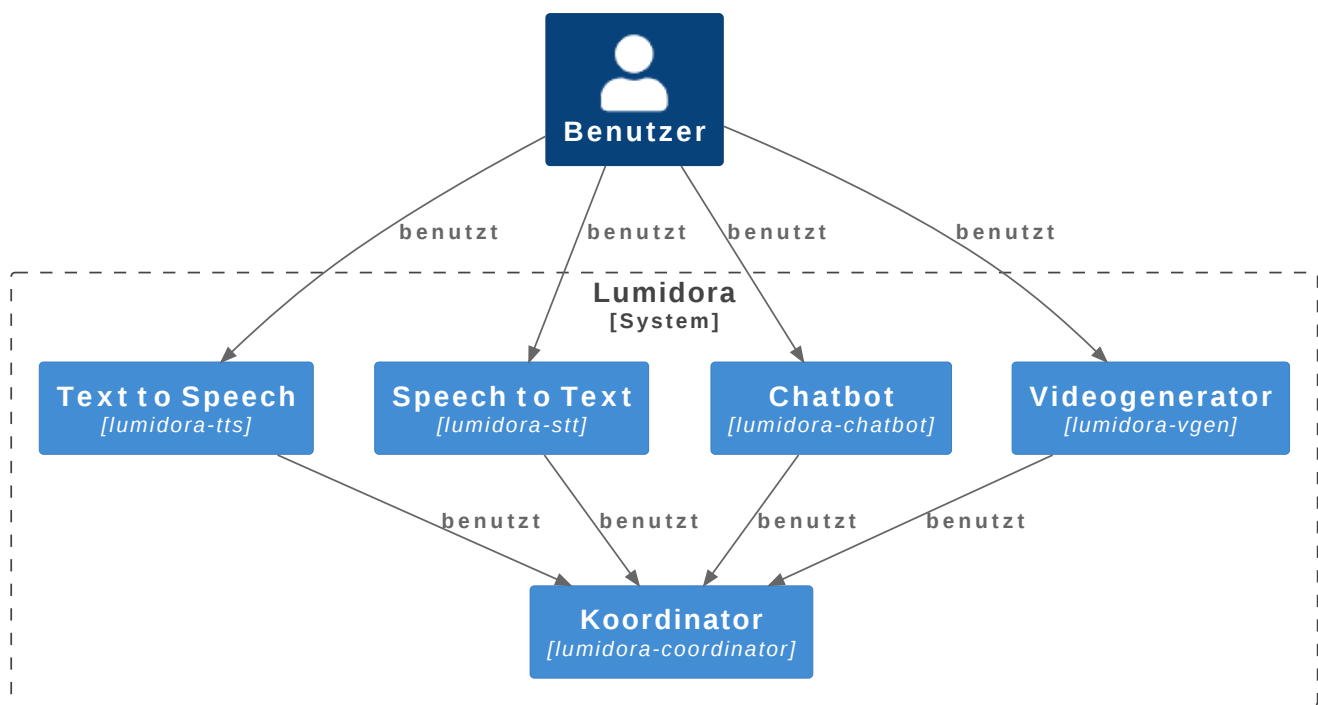
Systemkontext Technisch "Wie"



Bausteinansicht

Funktionale Sicht (Ebene 1)

Bausteindiagramm Funktional



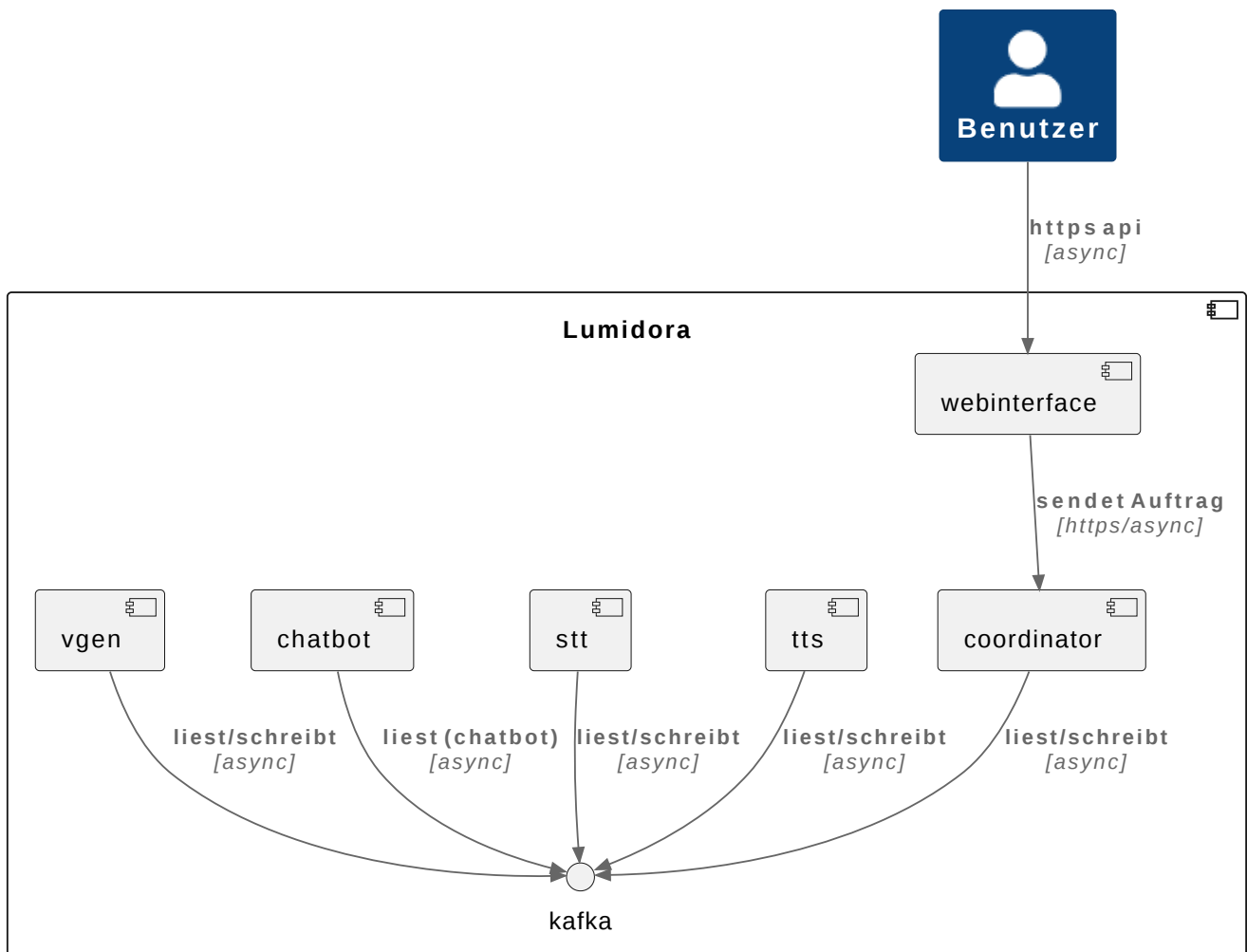
Legend

person
system
container
external person
external system
external container

Entität	Titel	Beschreibung
Benutzer	Benutzer des Systems	Verwendet Lumidora für seine Zwecke
lumidora-stt	Speech to Text	Wandelt Sprache in Text
lumidora-tts	Text to Speech	Wandelt Text in Sprache um
lumidora-chatbot	Chatbot	Herz des Systems. Verwendet alle anderen Systeme
lumidora-vgen	Videogenerierung	Generiert ein Video mithilfe einer Audiodatei und einem Portrait Bild.
lumidora-coordinator	Koordinator	Stellt die Kommunikation zwischen den einzelnen Komponenten sicher

Technische Sicht (Ebene 1)

Bausteindiagramm Technisch



Legend

person
system
container
component
external person
external system
external container
external component

Entität	Titel	Beschreibung
webinterface	Benutzeroberfläche	Wandelt Sprache in Text
stt	Speech to Text	Wandelt Sprache in Text
tts	Text to Speech	Wandelt Text in Sprache um
chatbot	Chatbot	Herz des Systems. Verwendet alle anderen Systeme
vgen	Videogenerierung	Generiert ein Video mithilfe einer Audiodatei und einem Portrait Bild.
kafka	Koordinator	Stellt die Kommunikation zwischen den einzelnen Komponenten sicher

Anforderungen

Funktional

Funktional

In diesem Abschnitt geht es um die Anforderungen, welche Lumidora erfüllen soll.

Kernaufgabe des Systems

Lumidora soll eine einfache und schnelle Verwendung von aktuellen Open Source AI Projekten ermöglichen. Er soll die Fähigkeiten dieser Projekte bündeln und schließlich ein digitaler Assistent und Content Erzeuger mithilfe dieser Tools werden. Er soll mindestens folgende Fähigkeiten haben

- Text to Speech
- Speech to Text
- Text to Image
- Audio to Movie
- Text 2 Image

Kategorie des Systems

Es handelt sich um ein System, welches hauptsächlich auf künstlicher Intelligenz beruht. Es soll nützliche Open Source AI Tools verknüpfen und dem Benutzer einen extremen Mehrwert bieten, sowohl im privaten, wie auch im Beruflichen. Das Tool soll in der Lage sein, verschiedenste Arten von Kontent möglichst autonom erzeugen zu können.

Stakeholder

Im weiten ist jeder Stakeholder, welcher sich von AI unterstützen lassen will. Im engeren Sinne, sind die Stakeholder, welche mit AI Kontent jeglicher Art in jeglicher Medialen Ausprägung (Audio, Video, Schriftlich, Bildlich) erstellen möchten.

Mögliche Stakeholder

- Content Creator
- Softwareentwickler
- Benutzer der mit dem PC arbeitet

anforderungen/funktional/anwendungsfaelle.adoc

=Anwendungsfälle

Nicht Funktional

Qualität

In diesem Abschnitt geht es um die Qualitätsanforderungen, welche an Lumidora gestellt werden. Um Qualitätsanforderungen strukturiert zu erheben, helfen uns Qualitätsmodelle. Ein weitverbreitetes Qualitätsmodell ist der ISO 25010-Standard.

Qualitätsmerkmal

“Qualitätsmerkmal - messbare, qualitätsbezogene Eigenschaften eines Systems. Zum Beispiel Fehlertoleranz, Ressourcenverbrauch, usw.

— oose
innovative Informatik eG

Top 3 Qualitätsmerkmale dieses Projektes

In diese Projekt legen wir unseren Fokus auf die folgenden 3 Qualitätsmerkmale:

1. Zuverlässigkeit
2. Stabilität und Fehlertoleranz
3. Benutzbarkeit

“Der ISO 25010 ist ein internationaler Standard, der Qualitätsmodelle und Qualitätsanforderungen für Softwareprodukte und -systeme definiert. Er definiert im wesentlichen 8 Qualitätsmerkmale: Effizienz (Performance), Wartbarkeit, Benutzbarkeit, Kompatibilität, Zuverlässigkeit, Sicherheit, Funktionale Eignung, Übertragbarkeit

Qualitätsziel

Aus den 3 wichtigsten Qualitätsmerkmalen leiten sich die 3 wichtigsten Qualitätsziele des Projektes ab.

“Ein Qualitätsziel ist eine Anforderung bzgl. eines Qualitätsmerkmals (Benutzbarkeit → Erlernbarkeit) , auf höherem Abstraktionsniveau. Zum Beispiel "Das System muss für die Angestellten unserer Personalabteilung leicht erlernbar sein."

— oose
innovative Informatik eG

Top 3 Qualitätsziele dieses Projektes

1. Lumidora muss eine hohe Ein- und Ausgabequalität liefern. (Zuverlässigkeit → Ausgereiftheit)
2. Lumidora muss Arbeitsaufträge die gestellt wurden erledigen können. (Stabilität und Fehlertoleranz)
3. Der Benutzer kann möglichst flexibel Inhalte generieren und die einzelnen Module von Lumidora ansprechen. (Benutzbarkeit)

“Ein Qualitätsziel kann mehrere Qualitätszenarien beinhalten

Qualitätsszenarien

“Ein Qualitätsszenario beschreibt, wie sich das System, wenn es gut implementiert wurde, auf bestimmte Impulse reagiert. Die Reaktion ist mess- oder beobachtbar

und macht ein Qualitätsmerkmal des Systems für die Akteure (siehe Systemkontext) oder andere Stakeholder erlebbar.

— oose
innovative Informatik eG

“Typische Bestandteile eines Qualitätsszenarios sind Quelle, **Auslöser**, **Umgebung**, Artefakt, **Antwort**, Antwort-Maß

— oose
innovative Informatik eG

Lumidora muss eine hohe Ein- und Ausgabequalität liefern.

1. Generierung eines YouTube Videos: Ein Benutzer (*Auslöser*) gibt dem ai-assistenten einen Text-Prompt zum Thema Klimawandel. (*Umgebung*) Der ai-assistent generiert eine Video zu diesem Thema, welches den eingegebenen Wünschen des Benutzers (Text-Prompt) entspricht_ und eine zufriedenstellende Qualität für den Benutzer darstellt. (*Antwort*) Dieses Ziel muss Benutzerunabhängig, maximal nach 10 Generierungsversuchen der Fall sein. (*Antwort-Maß*)
2. Spracheingabe in Text umwandeln: Ein Benutzer (*Auslöser*) nutzt die TTS- Funktion des ai-assistent und spricht durch ein durchschnittlich gutes Mikrofon in durchschnittlich gutem Deutsch und mit korrektem Abstand zum Mikrofon. Die Aufnahme wird nicht durch Hintergrundgeräusche beeinträchtigt. (*Umgebung*) Das Textergebnis entspricht dem gesprochenen Text (*Antwort*) . Minimale Abweichungen bei mehreren Sätzen sind erlaubt (*Antwort-Maß*).

Risiken

:navtitle:Risiken

Eines der entscheidenden Risiken ist, dass die erstellten Inhalte von Lumidora keine ausreichend hohe Qualität haben. Es wird vorausgesetzt, dass eingesetzte Open Source Tools, deren aktuelle Ein- und Ausgabequalität noch nicht ausreichend ist, in Zukunft ihre Qualität steigern können.

Randbedingungen

Organisatorisch

“Organisationen, die Systeme entwerfen, [...] sind gezwungen, Entwürfe zu erstellen, die die Kommunikationsstrukturen dieser Organisationen abbilden.

— Conways Law
How Do Committees Invent?

In diesem Abschnitt geht es um die Organisatorischen Randbedingungen, welche für die Anwendung lumidora gelten.

Organisation

Organisationsstruktur der Auftraggeber

Es handelt sich um ein privates Open Source Projekt. Daher muss die Organisationsstruktur nicht berücksichtigt werden und es gibt diesbezüglich keine Randbedingungen.

Eigene Organisationsstruktur

Keine eigene Organisationsstruktur da privates Open Source Projekt.

Kooperationspartner

Keine Kooperationspartner.

Vorhandenes Know How

Grundsätzliches Know How vorhanden. Lernwillig und motiviert mehr zu lernen.

Teamgröße

Initial eine One Man Show.

Teamaufteilung & Standorte

Keine Teamaufteilung notwendig aktuell, da nur eine Person daran entwickelt.

Verfügbarkeit des Teams

Privates Projekt, daher keine klare Aussage diesbezüglich möglich.

Ressourcen

Zeit

Privates Projekt, keine klare Zeitaufteilung möglich.

Geld

Kein Geld vorhanden.

Standards

Vorgehensmodell

Iterativ und inkrementell nach Scrum, Architekturvorgehen : Brezelvorgehen :-)

Entwicklungswerkzeuge

Keine Angaben.

Testwerkzeuge und -prozesse

Keine Angaben.

Abnahme- und Freigabeprozess

Keine Angaben.

Service Level Agreements

Keine Angaben.

Qualitätsstandards

Keine Qualitätsstandards, allerdings werden konkrete Qualitätsanforderungen im Abschnitt Qualitätsanforderungen später definiert.

Dokumentationsrichtlinien

Dokumentation nach Arc 42

Juristische Fragen

Grundsätzlich gibt es im AI Umfeld viele juristische Unklarheiten. Da es sich um ein Privates Projekt handelt und kein Juristisches Know How vorhanden ist, kann aktuell nicht darauf eingegangen werden. Hier wird während des Projektverlaufs versucht, die Juristischen Fragen zu beantworten.

Datenschutz

Haftungsfragen

Nachweispflicht

Internationale Rechtsfragen

Revisionssicherheit

Technisch

Technische Randbedingungen

In diesem Abschnitt geht es um die Randbedingungen, welche für die Anwendung lumidora gelten. Das Projekt soll nur auf Open Source Software beruhen und lokal, ohne Internet lauffähig sein.

Zielhardware

ddd

Prozessor, RAM

Da lumidora auch lokal laufen soll und ai Anwendungen hohe Hardwareanforderungen haben gibt es folgende Mindestanforderungen:

- Moderner CPU i9 9X
- NVidia Grafikkarte mit mindestens 8 GB Arbeitsspeicher
- Mindestens 16 GB Ram

lumidora wird auch auf weniger Hardware lauffähig sein, es wird jedoch davon abgeraten.

Netzwerk und Firewalls

Besondere Randbedingungen für Netzwerk und Firewall gelten nicht.

Softwarevorgaben

Betriebssystem

lumidora wird hauptsächlich für Linux entwickelt und soll schlussendlich in Docker-Containern lauffähig sein.

Datenbanken

Aktuell ist eine Zwischenspeicherung im ersten Iterationsschritt nicht angedacht. Vorstellbar ist dies aber in Zukunft um dem lumidora eine Art Gedächtnis zu verschaffen.

Middleware

Keine Middleware nötig

Frameworks

Es werden verschiedene Frameworks im Bereich AI zum Einsatz kommen. Ebenfalls werden mehrere Github Projekte inkludiert werden.

Proxies

Kein Proxy vorhanden

Caches

Kein Cache notwendig

Monitoringsystem

Auf ein Monitoringsystem kann zunächst verzichtet werden

Application Server

Ein Application Server wird zunächst nicht benötigt.

Webserver

Eine Benutzeroberfläche im Browser ist zunächst nicht angedacht. Daher wird zunächst kein Webserver benötigt.

ID-Managementsystem

Da es sich zunächst um eine Anwendung handelt, welche Lokal für einzelne Benutzer gedacht ist, wird zunächst keine Authentifizierung nötig sein.

Betriebliche Aspekte

On-Premise oder Cloud

Zunächst soll lumidora lokal lauffähig sein. Durch die Anforderung Dockerkompatibel zu sein, ist eine einfache On-Premise, bzw. Cloud Integration schnell möglich.

Online- oder Batchbetrieb

lumidora soll in Echtzeit mit dem Benutzer interagieren können, aber auch Aufgaben asynchron, bspl. über Nacht erledigen können. Daher soll sowhol ein Online, wie auch ein Batchbetrieb möglich sein.

Betriebskosten

Solange die Software lokal läuft, fallen keine Betriebskosten an, sondern lediglich die Initialen Hardwarekosten beim Kauf.

Verfügbarkeit / Support

Verfügbarkeit im Lokalen Umfeld nicht relevant. Wenn lumidora On-Premise oder in der Cloud betrieben werden soll, muss sich der jeweilige Betreiber darum kümmern.

Wartungsfenster

Keine Wartungsfenster notwendig.

Zugriffsmöglichkeiten

Folgende Zugriffsmöglichkeiten werden implementiert (Reihenfolge: Absteigend nach Wichtigkeit sortiert)

Programmatisch über Python

- Konsole
- Restschnittstelle
- Frontend Python
- Frontend Web

Konfigurationsmanagement

Ein Konfigurationsmanagement ist nicht notwendig.

Entwicklungsvorgaben

Programmiersprache

Der Kern wird in Python programmiert, da Python im AI Open Source Bereich die gängigste Programmiersprache ist. Für spätere Sekundärprogramme kann je nach Notwendigkeit auf andere Programmiersprachen ausgewichen werden.

Entwicklungsumgebung

Die Entwicklungsumgebung ist frei wählbar.

Protokolle, Buildserver, Buildpipeline

Keine Vorgaben.

GUI-Gestaltung

In Aktueller Iteration werden keine Vorgaben bzgl. GUI gemacht, da zunächst auf die Implementierung des Kerns fokus gelegt wird. Ist die Entwicklung des Kerns weitgenug vorangeschritten, wird darüber nachgedacht.

API

Die Api soll eine einfache und lose Kopplung zwischen den einzelnen Kernmodulen gewährleisten. Eine einfacher Austausch und Integration anderer Komponenten soll einfach möglich sein.

Namenskonventionen, Programmierrichtlinien, Versionsverwaltung

Sehen wir in diesem Projekt als Designvorgabe und verlagern die genaue Definition ins Entwicklerteam. Das Team soll, für alle beteiligten Entwicklern akzeptierte, Programmierrichtlinien für dieses Projekt iterativ erarbeiten.

Architekturentscheidungen

Open Source

Fragestellung

Was genau ist das Problem ?

Ein Ziel dieses Systems ist es, das es autark, ohne Internet und ohne fremde Dienste, nur mit Open Source Lösungen funktioniert. Da aktuell Open Source Alternativen noch nicht in allen Bereichen die Qualität bieten, welche Closed Source Alternativen bieten, wird befürchtet, dass eine zufriedenstellende Erstellungsqualität mit Open Source Lösungen alleine nicht erreicht werden kann.

Warum ist es für die Architektur relevant ?

Zentrale Bausteine des lumidora sind externe Open Source Projekte und Lösungen. Stellt sich zu einem späteren Zeitpunkt heraus, dass einige, speziell der Chatbot, keine ausreichend gute Qualität liefern, wenn sie mit Open Source Lösungen umgesetzt werden, ist dies Projektentscheidend.

Welche Auswirkungen hat die Entscheidung ?

Sie ist Projektentscheidend, da die wichtigste Eigenschaft des lumidora ist, nützlichen, qualitativ hochwertigen Output für den Benutzer zu liefern.

Einflussfaktoren

Welche Randbedingungen haben wir einzuhalten

Das Projekt soll nur auf Open Source Software beruhen und lokal, ohne Internet lauffähig sein.

Welche Qualitätsziele sind zu beachten

Hohe Ein- und Ausgabequalität der generierten Inhalte

Welche Risiken sind betroffen

Das Risiko, dass die Ein- und Ausgabequalität nicht ausreichend ist.

Annahmen

Welche Annahmen haben wir getroffen

- Closed Source Alternativen sind aktuell in einigen Bereichen den Open Source Lösungen überlegen
- Wir gehen davon aus, dass sich das Blatt im Laufe der Zeit ändern wird und Open Source Lösungen mindestens zu den Closed Source Lösungen gleichwertig sind.

Welche Annahmen können vorab wie überprüft werden

Es ist möglich, die einzelnen Sprachmodelle zu testen. Es gibt auch Vergleiche, die beispielsweise zeigen, dass das Closed Source Projekt ChatGPT 4 bis jetzt noch weit besser ist als vergleichbare Open Source Modelle. Wir können nicht überprüfen, ob in Zukunft Open Source Sprachmodelle mindestens gleichwertig sind.

Mit welchen neuen Risiken müssen wir rechnen

Nur das oben bereits erwähnte Risiko, das die Ein- und Ausgabequalität nicht zufriedenstellend ist, weil Open Source Modelle nicht die Qualität erreichen wie vergleichbare Closed Source Modelle.

Alternativen

Welche Lösungsoptionen ziehen wir in die nähere Auswahl ?

Wir konnten mehrere Lösungsoptionen identifizieren

- Verwenden von Closed Source Alternativen in den Bereichen, indem Open Source aktuell noch besser ist.
- Verwenden von Open Source Projekten auch wenn sie aktuell noch schlechter sind.
- Dauerhafter Einsatz von Closed Source Projekten, wenn diese für ein speziellen Bereich besser sind.

Wie bewerten wir jede einzelne ?

Welche Option schließen wir bewusst aus ?

Die letzte Option fällt direkt weg, da sie gegen einer unser wichtigsten Grundsätze verstößt und wir keine Closed Source Produkte verwenden wollen und unser assistent offline laufen soll. Bleiben die ersten beiden Optionen.

Ergebnis

Wir nehmen immer Open Source Tools, auch wenn die Ein- und Ausgabequalität aktuell noch schlechter ist als bei Closed Source Alternativen.

Wer hat die Entscheidung getroffen ?

Dominik Bruhn

Wie ist sie begründet ?

Aufwand vor Ein- und Ausgabequalität, da das Projekt aktuell nur von einer Person gemacht wird.

Wann wurde entschieden ?

Am 02.11.2023